Automatica 186 (2026) 112767

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

# On dissipativity of cross-entropy loss in training ResNets — A turnpike towards architecture search ☆

Jens Püttschneider [a], Timm Faulwasser [b,*]

[a] Institute of Energy Systems, Energy Efficiency and Energy Economics, TU Dortmund University, 44227 Dortmund, Germany
[b] Institute of Control Systems, Hamburg University of Technology, 21079 Hamburg, Germany

ARTICLE INFO

*Article history:*
Received 28 March 2024
Received in revised form 9 October 2025
Accepted 5 November 2025

*Keywords:*
Optimal control
Dissipativity
Deep learning
Neural networks
Manifold turnpike
Label smoothing
Huber loss

ABSTRACT

The training of ResNets and neural ODEs can be formulated and analyzed from the perspective of optimal control. This paper proposes a dissipative formulation of the training of ResNets and neural ODEs for classification problems. Specifically, we consider a variant of the cross-entropy (label smoothing) as a loss function *and* as a regularization in the stage cost. Based on our dissipative formulation of the training, we prove that the training OCPs for ResNets and neural ODEs alike exhibit the turnpike phenomenon. We illustrate this finding with numerical results for the two spirals and MNIST datasets. Crucially, our training formulation ensures that the transformation of the data from input to output is achieved in the first layers. In the following layers, which constitute the turnpike, the data remains at an equilibrium state and therefore these layers do not contribute to the transformation learned. In principle, these layers can be pruned after training, resulting in a network with only the necessary number of layers thus simplifying tuning of hyperparameters.



## 1. Introduction

Deep learning (DL) and (optimal) control theory share many interesting connections. For example, the seminal backpropagation algorithm for neural networks, is closely linked to the adjoint equation of optimal control (Bryson & Denham, 1962; Chen et al., 2018) and to backwards algorithmic differentiation (Griewank, 2012; Speelpenning, 1980).[1]

Moreover, the training of Neural Networks (NNs) with constant width in each layer can be formulated as an Optimal Control Problem (OCP) (Esteve et al., 2020; Li et al., 2017). In this context, the layer-to-layer propagation of the data is considered a dynamical system on a finite horizon corresponding to the depth of the network. The solution to the OCP determines the weights and biases of the neural network, i.e., weights and biases are the control inputs to drive the data to a desired point in the terminal layer determined by the label and loss function. The system and control perspective is particularly helpful for Residual Neural Networks (ResNets) (He et al., 2016b) which can be regarded as Euler forward discretizations of neural ODEs (Chen et al., 2018). Chang et al. (2018) analyze the reversibility and stability of the ResNet dynamics based on their continuous time counterparts. Alternatively, NN training can also be approached as a state estimation problem, wherein, the weights and biases follow the dynamics induced by the stochastic gradient methods and are observed from transient information about the loss (Bemporad, 2023; Singhal & Wu, 1988).

In the usual ResNet setting (He et al., 2016b), the objective of the training OCP is given by the loss function evaluated for the output of the neural network. From the optimal control point of view, this corresponds to a terminal penalty or a Mayer term. Additionally, a regularization term for neural network parameters is often considered which corresponds to a stage cost depending solely on the input. In Model Predictive Control (Rawlings et al., 2019), however, the terminal penalty of OCPs is often chosen as an approximation of the infinite horizon value function for a given stage cost. Esteve et al. (2020) and Faulwasser et al. (2024) have suggested to include a stage cost term based on the states of the hidden layers in the training OCP. Esteve and Geshkovski (2023) show that including the cross-entropy loss and $L_1$ parameter regularization in the stage cost for training a neural ODE leads to finite-time convergence of the state trajectory but not to classic turnpikes. Such hidden-state penalties also arise in practical machine learning contexts. For instance, joint training formulations for early exiting make use of auxiliary loss

☆ The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Subhrakanti Dey under the direction of Editor Florian Dorfler. This work has been partly funded by the German Federal Ministry of Research, Technology and Space (BMFTR) via the project 6GEM under funding reference 16KISK038.

* Corresponding author.
*E-mail addresses:* jens.puettschneider@tu-dortmund.de (J. Püttschneider), timm.faulwasser@ieee.org (T. Faulwasser).

[1] In 2024, John J. Hopfield and Geoffrey E. Hinton received the Nobel Prize in Physics for foundational discoveries and inventions that enable machine learning with artificial neural networks.

https://doi.org/10.1016/j.automatica.2025.112767

functions attached to intermediate layers (Rahmath et al., 2024; Teerapittayanon et al., 2016).

Specifically, Faulwasser et al. (2024) analyze dissipativity of ResNet training with quadratic ($\ell_2$) stage cost regularization. They show that the solutions to the training OCP exhibit the so-called turnpike phenomenon. These results imply that the learned transformation occurs after a finite number of layers which allows to constructively bound the depth required for a given learning task. The remaining layers are spent at an equilibrium state representing an identity mapping. Esteve et al. (2020) investigate turnpikes in the training of neural ODEs by analyzing OCP optimality conditions. Moreover, Ruiz-Balet and Zuazua (2023) establish reachability properties for neural ODEs with the ReLU activation function.

The role of dissipativity in deep learning is also analyzed by Feng and Lam (2011) and Zeng et al. (2015) for cellular neural networks, by Revay et al. (2023) for recurrent equilibrium networks, and by Martinelli et al. (2023) for neural ODEs. In these works a quadratic supply rate is considered. Moreover, Hamiltonian NNs address vanishing and exploding gradients during training using passivity (Galimberti et al., 2023).

In a nutshell, the turnpike phenomenon is a similarity property of optimal solutions for varying initial condition and varying horizon length (Faulwasser & Grüne, 2022; Faulwasser et al., 2014). The concept originated in optimal control approaches to economics (Dorfman et al., 1958; McKenzie, 1976) and early observations are due to Ramsey (1928) and von Neumann (1938). In the analysis of turnpike properties in optimal control there has been recent progress along two avenues: analysis of the optimality system (Sakamoto & Zuazua, 2021; Trélat & Zuazua, 2015) and leveraging dissipativity properties of the OCP (Damm et al., 2014; Faulwasser et al., 2017; Grüne & Müller, 2016). Interestingly, the dissipativity route is linked to the foundational work by Willems (1971) on infinite-horizon least-squares optimal control but it also generalizes to other objectives and nonlinear systems (Faulwasser & Kellett, 2021). The turnpike can be regarded as the attractor of the infinite horizon optimal solutions (Faulwasser & Kellett, 2021; Trélat, 2023). We refer to Grüne (2022) and to Faulwasser and Grüne (2022) for literature overviews. A recent trend in dissipativity-based analysis of OCPs is the generalization of turnpike properties towards more general turnpike objects such as subspaces (Schaller et al., 2021) or manifolds (Faulwasser et al., 2022; Karsai, 2024).

In this paper, we leverage subspace turnpike concepts to analyze ResNet training from an optimal control perspective. In contrast to the quadratic regularization costs proposed by Esteve et al. (2020) and Faulwasser et al. (2024), we consider a variant of the cross-entropy for classification tasks to obtain a dissipative formulation of ResNet training for classification tasks. Specifically, using the cross-entropy with label smoothing (or soft cross-entropy) we derive locally a suitable quadratic lower bound to the loss function. We also show that the soft cross-entropy behaves similar to *Huber loss* (Huber, 1992), i.e. locally quadratic with linear asymptotics. This lower bound allows to show strict dissipativity of our training OCP formulation with respect to the linear subspace of steady minimizers of the soft cross-entropy. We prove the existence of subspace turnpikes in the underlying training problem. Moreover, we extend our result to continuous-time formulations with neural ODEs and we propose sufficient conditions which enable extension to other NN architectures derived via implicit or explicit discretization of neural ODEs. The dissipativity-inducing stage cost based on the loss function encourages the network to learn the input–output transformation in its first layers. Evaluating the loss function for the states of the hidden layers then provides an indication of the performance of shallower networks which can be used in the search of NN architectures (Elsken et al., 2019). We illustrate the benefits of the proposed training formulation considering the two-spirals and the MNIST datasets.

The remainder of the paper is structured as follows: Section 2 introduces the optimal control formulation of deep learning. Section 3 provides the dissipative formulation of neural network training. Section 4 then uses the dissipative formulation to prove the existence of turnpikes in the trained NN. Section 5 extends the dissipative training formulation to neural ODEs. Section 6 validates the formulation by training a ResNet on the two spirals dataset and MNIST. We end with a conclusion and an outlook in Section 7.

## 2. Optimal control and ResNet training

The training of NN and neural ODEs in deep learning can be cast as an optimal control problem, where the control inputs are the network parameters that steer the data through the layers towards a representation in the last layer corresponding to the label. The perspective of optimal control is particularly beneficial for ResNets, which can be interpreted as Euler forward discretizations of neural ODEs.

The propagation of a data point $x^i$ through the layers of a ResNet can be conceptualized as a discrete time dynamical system of the form

$$x^i_{k+1} = x^i_k + \sigma\left(A_k x^i_k + b_k\right), \quad x^i_0 = x^i \in \mathbb{R}^C, \tag{1}$$

where the time step $k \in \mathbb{N}_{[0,N-1]}$ corresponds to the index of the residual layer in the $N$-layer network. When training the ResNet, the parameters, weights $A_k$ and biases $b_k$, are optimized to best fit the training data. Throughout this paper, the scalar and continuous activation function $\sigma : \mathbb{R} \to \mathbb{R}$ is applied element-wise to the activation vector $A_k x^i_k + b_k$. Moreover, we require $\sigma(0) = 0$ such that ResNets can render each state $x$ a steady state by choosing $A = 0$ and $b = 0$. This implies ResNets can learn identity mappings, preserving information from one layer to the next and thus often show superior performance of deeper ResNets over shallower ones (He et al., 2016a).[2]

The data propagated through the network corresponds to the state $x^i_k$, its initial condition is a feature $x^i$ from the dataset

$$\mathbb{D} = \left\{\left(x^1, y^1\right), \ldots, \left(x^D, y^D\right)\right\}.$$

The label $y^i$ determines the target of the state propagation. For classification tasks with $C$ different classes, the label $y^i \in \mathbb{Y} = \mathbb{N}_{[1,C]}$ represents the class index.[3] The goal of NN training is to learn a predictor $\hat{f} : \mathbb{R}^C \times \mathbb{R}^{n_w} \to \mathbb{Y}$, $\hat{y} = \hat{f}(x, w)$ with parameters $w \in \mathbb{R}^{n_w}$. The selection of hyperparameters that define the neural network architecture, such as its depths $N \in \mathbb{N}$ and activation function $\sigma(\cdot)$ is a crucial design decision that aims at finding the neural network structure best suited for the given training problem.

Closely related to ResNets are neural ODEs that provide a continuous-time formulation of deep learning

$$\dot{x}^i(t) = \sigma\left(A(t)x^i(t) + b(t)\right), \quad x^i(0) = x^i \in \mathbb{R}^C, \tag{2}$$

[2] In this paper, we refer to the full dynamics (1) evaluated at some step $k$ as *residual layer*. However, in the machine learning literature the term *residual block* is also used, whereby the notion *layer* is reserved for the mapping $x_k \mapsto A_k x^i_k + b_k$ (He et al., 2016a).

[3] One could formally introduce an output equation in (1) which resembles the functionality of an output layer in the ResNet. The parameters of the output layer can also be optimized for in the training process. Which, in the language of optimal control, leads to additional degrees of freedom. In view of our later analysis in Section 3, we simplify the exposition by not explicitly detailing the output layer. Our later analysis could easily consider this by adding extra decision variables to the loss function.

for a suitable discretization of the continuous parameters $A(t)$ and $b(t)$. In particular, fixed-step size Euler forward discretization of the neural ODE above yields the ResNet from (1). Henceforth, we will first consider the ResNet architecture before we turn to NODEs in Section 5.

### 2.1. Optimal control formulation of deep learning

To formalize the training on the entire dataset, we stack the individual data points

$$\mathbf{x}^0 \doteq \left[x^{1\top}, \dots, x^{D\top}\right]^\top \quad \mathbf{y} \doteq \left[y^1, \dots, y^D\right]^\top, \tag{3}$$

which allows writing the data as $\mathbb{D} = \{(\mathbf{x}^0, \mathbf{y})\}$. For the stacked data, the ResNet dynamics are

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \sigma\left((I^D \otimes A_k)\mathbf{x}_k + (\mathbf{1}^D \otimes b_k)\right) \\ &\coloneqq \mathbf{f}_\mathrm{d}(\mathbf{x}_k, u_k), \qquad \mathbf{x}_0 = \mathbf{x}^0 \in \mathbb{R}^{D\cdot C}, \end{aligned} \tag{4}$$

where $u_k = (\mathrm{vect}(A_k)^\top, b_k^\top)^\top \in \mathbb{R}^{n_u}$ is the vectorized control input of layer $k$ with dimensionality $n_u = C^2 + C$, $I^D \in \mathbb{R}^{D\times D}$ is the identity matrix, $1^D \in \mathbb{R}^D$ the vector of all ones, and $\otimes$ refers to the Kronecker product. Note that as per (3), the boldface state variable $\mathbf{x}$ indicates the stacked state for all data samples, which are in this formulation simultaneously controlled by one ResNet, i.e., one input sequence $u_k$ which entails the weights and biases. As a shorthand, we write state-input pairs as $\mathbf{z} = (\mathbf{x}, u)$.

For a number of residual layers $N$ (i.e. for a network depth N), training the ResNet can be formulated as the discrete-time OCP

$$V_N^\gamma\left(\mathbf{x}^0\right) = \min_{u_{0,\dots,N-1}} \sum_{k=0}^{N-1} \ell(\mathbf{x}_k, u_k) + \gamma\,\ell_\mathrm{f}(\mathbf{x}_N, \mathbf{y}) \tag{5a}$$

$$\text{s.t.} \quad \mathbf{x}_{k+1} = \mathbf{f}_\mathrm{d}(\mathbf{x}_k, u_k) \quad \forall k \in \mathbb{N}_{[0,N-1]}, \tag{5b}$$

$$\mathbf{x}_0 = \mathbf{x}^0 \in \mathbb{R}^{C\cdot D}, \tag{5c}$$

where the Mayer term (terminal penalty) $\ell_\mathrm{f} : \mathbb{R}^C \times \mathbb{Y} \to \mathbb{R}_0^+$ is the loss function applied to the entire dataset and describes the quality of the NN output. The scalar $\gamma \in \mathbb{R}_0^+$ is used to trade-off the importance of the regularization against the loss function at the terminal layer. Typically, one uses the empirical loss, averaging the loss over the dataset for which, with slight abuse of notation,

$$\ell_\mathrm{f}(\mathbf{x}_N, \mathbf{y}) \doteq \frac{1}{D}\sum_{i=1}^{D} \ell_\mathrm{f}(x_N(x^i), y^i).$$

Here, as before the boldface letter arguments $\mathbf{x}$ and $\mathbf{y}$ indicate the dependence on the entire dataset, whereas $\ell_\mathrm{f}(x_N(x^i), y^i)$ is the loss of the individual data sample $(x^i, y^i)$. Additionally, the stage cost $\ell : \mathbb{R}^{C\cdot D} \times \mathbb{R}^{n_u} \to \mathbb{R}_0^+$ captures the regularization terms. These include the input regularization, a penalty on the parameters of the neural network $\ell(\mathbf{x}, u) = \|u\|_p^p$ for either $p = 1, 2$.[4]

In contrast to standard NN training formulations, we also allow the stage cost to explicitly depend on the states of the hidden layers, thus the loss function evaluated for the hidden layers can be included in the training objective. From a ML perspective one could thus refer to the full training objective as the loss function. Henceforth, we only refer to $\ell_\mathrm{f}$ as the loss function. Particular choices of the stage cost will be introduced later.

Using standard tricks of optimal control (Chachuat, 2007), it is indeed possible to reformulate OCP (5), which is in Bolza form, and thus include a stage cost and terminal penalty, into an OCP in Mayer form which only includes a terminal penalty. Doing so would require to add an additional state variable which captures the summation over the stage cost terms. Our later dissipativity analysis is, however, simplified in the chosen formulation (5).

The solution to OCP (5) are the NN parameters, weights and biases, i.e. the control inputs denoted as $u^\star(\mathbf{x}^0)$ and the resulting ensemble data trajectories $\mathbf{x}^\star\left(\mathbf{x}^0\right)$, which depend on the dataset as highlighted by $\mathbf{x}^0$. From the optimal control perspective, the main difficulty of NN training lies in the simultaneous control of $D$ data samples with only one network, i.e. only one control signal is applied to many initial conditions.

### 2.2. Cross-entropy loss for classification

In classification, the goal is to predict a discrete class $y$ given the feature $x$ (LeCun et al., 2015). For the loss function this means comparing the continuous state to the discrete class label. This is typically done by the cross-entropy loss function first proposed by Cox (1958). The cross-entropy first calculates probabilities for all possible classes $y \in \mathbb{Y}$ from the NN output state $x$ using the softmax activation function

$$p(y|x) = \frac{e^{[x]_y}}{\sum_{i=1}^{C} e^{[x]_i}},$$

whereby the operator $[\cdot]_i$ accesses the $i$th component of a vector. The predicted class $\hat{y} = \arg\max_i p(i|x)$ is the class with the highest probability. The probabilities for all classes are then arranged into the probability vector

$$p(x) = \left[p(1|x), \dots, p(C|x)\right]^\top. \tag{6}$$

Likewise to (6), we define the vector of target probabilities $q(y)$ determined by the label $y$. Typically binary targets $q(i|y) = \delta_{y,i}$ are used, where the probability of the labeled class $y$ is one and zero for all other classes.

Then, the output probability distribution vector $p(x)$ is compared to the target vector $q(y)$ induced by the labels using the cross-entropy leading to the loss function

$$\ell_\mathrm{f}(x, y) = -H(p(x), q(y)) = -\sum_{i=1}^{C} q(i|y)\log p(i|x).$$

For the binary targets, the cross-entropy only depends on the softmax probability of the correct class

$$\ell_\mathrm{f}(x, y) = -\log p(y|x). \tag{7}$$

### 2.3. Dissipativity of OCPs

We recall the definition of dissipativity of OCPs (Angeli et al., 2012), based on the dissipativity notion for open dynamical systems coined by Willems (1972). Moreover, we use an extended definition of dissipativity with respect to a set of optimal steady state pairs $\bar{\mathbb{Z}}^\star$, similar to Martin et al. (2019) and Müller (2021). The set of steady state pairs is given by

$$\bar{\mathbb{Z}} = \left\{\bar{\mathbf{z}} = (\bar{\mathbf{x}}, \bar{u}) \in \mathbb{R}^{D\cdot C} \times \mathbb{R}^{n_u} \mid \bar{\mathbf{x}} = \mathbf{f}_\mathrm{d}(\bar{\mathbf{x}}, \bar{u})\right\} \tag{8}$$

Optimal steady state pairs are computed via

$$\bar{\mathbf{z}}^\star \in \arg\min_{\bar{\mathbf{z}}} \ell(\bar{\mathbf{z}}) \quad \text{s.t.} \quad \bar{\mathbf{z}} \in \bar{\mathbb{Z}} \tag{9}$$

and the set of all optimal steady states is written as $\bar{\mathbb{Z}}^\star \subseteq \bar{\mathbb{Z}}$.

Recall that the distance between a point $x \in \mathbb{R}^{n_x}$ and the closed set $\mathbb{X} \subset \mathbb{R}^{n_x}$ is given by

$$\mathrm{dist}\,(x, \mathbb{X}) \doteq \min_{x' \in \mathbb{X}} \|x - x'\|.$$

A continuous function $\alpha : \mathbb{R}^+ \to \mathbb{R}^+$ is said to be a class $\mathcal{K}$ function, $\alpha \in \mathcal{K}$, if $\alpha(0) = 0$ $\alpha$ is strictly increasing. If, in addition, $\lim_{x\to\infty} \alpha(x) \to \infty$, then $\alpha \in \mathcal{K}_\infty$.

[4] We remark that in the ML literature also the term weight decay is used.

**Definition 1** (*Strict Dissipativity in Discrete Time*)**.** The discrete time dynamical system (4) is said to be dissipative with respect to a set of steady-state pairs $\bar{\mathbb{Z}}^\star$ if there exists a non-negative storage function $\lambda : \mathbb{R}^{D\cdot C} \to \mathbb{R}_0^+$ such that for all $\mathbf{z} = (\mathbf{x}, u)$ and all $\bar{\mathbf{z}}^\star \in \bar{\mathbb{Z}}^\star$

$$\lambda(\mathbf{f}_\mathrm{d}(\mathbf{z})) - \lambda(\mathbf{x}) \leq \ell(\mathbf{z}) - \ell(\bar{\mathbf{z}}^\star). \tag{10a}$$

If additionally, there exists $\alpha_\ell \in \mathcal{K}$ such that

$$\lambda(\mathbf{f}_\mathrm{d}(\mathbf{z})) - \lambda(\mathbf{x}) \leq \ell(\mathbf{z}) - \ell(\bar{\mathbf{z}}^\star) - \alpha_\ell\left(\mathrm{dist}\left(\mathbf{z}, \bar{\mathbb{Z}}^\star\right)\right), \tag{10b}$$

then the system (4) is said to be strictly $\mathbf{x}$-$u$ dissipative with respect to $\bar{\mathbb{Z}}^\star$ and for $\mathbf{z}$ replaced by $\mathbf{x}$ in the class $\mathcal{K}$ function the system is said to be strictly $\mathbf{x}$ dissipative.

OCP (5) is said to be (strictly) $\mathbf{x} - u$ dissipative with respect to $\bar{\mathbb{Z}}^\star$ if for all $N \in \mathbb{N}$ and all $\mathbf{x}_0 \in \mathbf{X}_0$, the dissipation inequalities 1 hold along any optimal trajectory of (5). □

For a singleton set $\bar{\mathbb{Z}}^\star = \{\bar{\mathbf{x}}^\star; \bar{u}^\star\}$ the above definition corresponds to the standard dissipativity notion with respect to an optimal steady state $\bar{\mathbf{x}}^\star$. Notice that, in view of Definition 1, the dissipativity of OCP (5) only depends on the regularization $\ell(\mathbf{x}, u)$ and not on the loss function, i.e. the Mayer term $\ell_\mathrm{f}$. This implies that the design of dissipative formulations for NN training needs to focus on the stage cost, while in deep learning the focus is usually on the choice of the loss functions as Mayer terms (Wang et al., 2020).

## 3. Dissipativity of cross-entropy loss in ResNets

Next, we turn towards analyzing the dissipativity properties of OCP (5). As a preparatory step we investigate the minimization properties of cross-entropy and its variant the soft cross-entropy.

### 3.1. Conceptual difficulties of standard cross-entropy

Consider the stage cost

$$\ell(\mathbf{x}, u) = \ell_\mathrm{f}(\mathbf{x}, \mathbf{y}) + r\|u\|^2, \tag{11}$$

in the training OCP (5).

**Lemma 2** (*No Finite Minimizers of Cross-Entropy*)**.** *The stage cost (11), which includes the cross-entropy with binary target probabilities, has no unconstrained minimizers in* $\mathbb{R}^{C\cdot D} \times \mathbb{R}^{n_u}$.

**Proof.** The stage cost (11) only depends on the control input $u$ in its input regularization $r\|u\|^2$, which is minimized if $u = 0$. The stage cost depends on $\mathbf{x}$ through the loss function $\ell_\mathrm{f}(\mathbf{x}, \mathbf{y})$ which is the averaging of the loss function for the individual and independent datasamples. The first-order necessary conditions of optimality for minimizing the cross-entropy with binary target probability for a single datasample $(x^i, y^i)$ reads

$$\frac{\partial \ell_\mathrm{f}(x^i, y^i)}{\partial x^i} = p(x^i) - q(y^i) = 0.$$

The softmax probability vector $p(x^i)$ thus has to match the vector of binary target probabilities. This implies for its $y$-th component that

$$p(y|x^i) = \frac{e^{\left[x^i\right]_y}}{\sum_{j=1}^C e^{\left[x^i\right]_j}} = \begin{cases} 1 & \text{if } y = y^i \\ 0 & \text{if } y \neq y^i \end{cases}.$$

However, for any finite value of $x^i$ the softmax probabilities cannot attain values of exactly zero or one. Hence, the cross-entropy with binary target probabilities attains the value 0 only asymptotically, i.e. if $\left[x^i\right]_j \to \infty$ if $j = y_i$ and $\left[x^i\right]_j \to -\infty$ if $j \neq y_i$. In other words, the states are pushed to infinity/minus infinity, while the infimum of the loss function on $\mathbb{R}^{D\cdot C} \times \mathbb{R}^{n_u}$ is $\ell(z^\star) = 0$. ■

This result implies that the strict dissipation inequality (10b) cannot hold with $\alpha_\ell \in \mathcal{K}_\infty$, i.e., (10b) does not hold if $\alpha_\ell$ is radially unbounded. Another consequence of the cross-entropy attaining its infinimum if the states diverge to infinity is that classic turnpike analysis concepts, which rely on reachability assumptions break down. The time the optimal solutions can spend far away from the infinite states is not bounded independent of the horizon length. Indeed, for any finite horizon in OCP (5) the solutions are always arbitrarily far away from the infinite states for which the cross-entropy attains its infimum. As we see later this contradicts the classic definition of the measure turnpike property as used in Section 4. Subsequently, instead of changing the turnpike concepts and as NN of finite depth are more application relevant, we adapt the considered loss function.

### 3.2. Soft cross-entropy and its properties

Label smoothing, first introduced by Szegedy et al. (2016), utilizes a target probability reachable by the softmax activation function. For a sample from class $y$, the target probability for class $i$

$$\tilde{q}(i|y) = \begin{cases} p_d & i = y \\ \frac{1-p_d}{C-1} & i \neq y \end{cases}, \tag{12}$$

is used. The main motivation of the soft cross-entropy is robustification of classification tasks with label noise, i.e. uncertainties in the labels. These occur if some of the labels observed from the dataset do not match the actual label, this can be due to human errors in the labeling process or due to ambiguity in the classification task itself, e.g., multiple classes are present in the same image. In these situations the soft cross-entropy has produced better generalization properties in empirical studies (Müller et al., 2019). To prevent overconfident false classification on samples, the true class is assigned a probability $p_d$ close to one, while the remaining non-zero probability is shared uniformly between the $C - 1$ incorrect classes.

The soft cross-entropy, i.e. the cross-entropy between the softmax probabilities $p(x)$ and the smoothed target distribution $\tilde{q}(y)$, is then used as the loss

$$\begin{aligned} \tilde{\ell}_\mathrm{f}(x, y) &= -H(p(x), \tilde{q}(y)) - \tilde{\ell}^\star \\ &= -\sum_{i=1}^C \tilde{q}(i|y) \log p(i|x) - \tilde{\ell}^\star. \end{aligned} \tag{13}$$

In this work, we use the constant offset $\tilde{\ell}^\star = -H(\tilde{q}(y), \tilde{q}(y))$, such that the minimum of the soft cross-entropy is zero. To analyze the dissipativity of the training using the soft cross-entropy we need to find its stationary minimizer.

**Lemma 3** (*Minimizers of Soft Cross-Entropy*)**.** *Consider the minimization of the soft cross-entropy (13) for the label $y$*

$$\mathbb{X}_y^\star = \arg\min_{x_y \in \mathbb{R}^C} \ \tilde{\ell}_\mathrm{f}(x_y, y).$$

*The minimizers form the line*

$$\mathbb{X}_y^\star = \left\{x \in \mathbb{R}^C \,\middle|\, [x]_c = [x]_y + \delta,\ \forall c \in \mathbb{Y} \setminus \{y\},\ \forall [x]_y \in \mathbb{R}\right\},$$

*where* $\delta = -\log\left(\frac{(C-1)p_d}{1-p_d}\right)$. *Moreover, the corresponding minimum value*

$$\min_{x \in \mathbb{R}^C} \ \tilde{\ell}_\mathrm{f}(x, y) = \tilde{\ell}^\star - \tilde{\ell}^\star = 0,$$

*is independent of the label $y$.* □

**Proof.** The soft cross-entropy is the cross-entropy of the softmax probabilities $p(x)$ and the smoothed target probabilities $\tilde{q}(y)$ $\tilde{\ell}_{\mathrm{f}}(x,y) = -H(p(x),\tilde{q}(y)) - \tilde{\ell}^\star$. The cross-entropy $H(p,q)$ becomes minimal if $p = q$ (Goodfellow et al., 2016, Section 3.13), and hence the soft cross-entropy becomes minimal if $p(x) = \tilde{q}(y)$. This implies

$$p(c|x^\star) = \frac{e^{[x^\star]_c}}{\sum_{i=1}^{C} e^{[x^\star]_i}} = \begin{cases} p_d & c = y \\ \frac{1-p_d}{C-1} & c \neq y. \end{cases}$$

The equations for the incorrect classes, $p(c \neq y|x)$, only differ in the numerator $e^{[x^\star]_{c\neq y}}$, such that all $[x^\star]_{c\neq y}$ have to be equal. Dividing $p(y|x)$ by $p(c \neq y|x)$ gives $\frac{e^{[x^\star]_y}}{e^{[x^\star]_{c\neq y}}} = \frac{(C-1)p_d}{1-p_d}$. We obtain the line, $\forall c \in \mathbb{Y} \setminus \{y\}, \forall [x^\star]_y \in \mathbb{R}$

$$[x^\star]_c = [x^\star]_y - \log\left(\frac{(C-1)p_d}{1-p_d}\right)$$

which forms $\mathbb{X}_y^\star$. ■

**Remark 4** (*Large Data with* $\dim x > C$)**.** For many datasets, such as MNIST (Deng, 2012), the dimensionality of the input features defines the input layer size. The width of the output layer meanwhile has to match the number of classes $C$ for the softmax activation function. Thus, the network architectures used for these datasets have progressively narrower layers (Krizhevsky et al., 2012; Lecun et al., 1998). In contrast, the system-theoretic analysis is simplified if the width does not change across layers.

A simple solution is to keep the state dimension of the state equal to the input features and only consider the $C$-first (or $C$ specified) components of in the loss function. Then, the remaining components of the state $[x]_{C+1,\dots,n}$ do not contribute to the loss nor the distance to $\mathbb{X}_y^\star$. Without loss of generality, our analysis is done only for the case that the dimension of the state matches the number of classes $\dim x = C$. Indeed, our main dissipation and turnpike results leverage the geometry of the soft cross entropy and thus also hold for the case that $n > C$.

Moreover, an output layer could be included to map $x$, $\dim x = n$, to an output $y_{\mathrm{p}}$ predicting the label $y$ with $\dim y_{\mathrm{p}} = C$. This may potentially induce additional decision variables in the training problem, cf. Footnote 3 □

Now, we introduce several technical lemmas relating $\tilde{\ell}_{\mathrm{f}}(x,y)$ to the distance to the set of it minimizers $\mathbb{X}_y^\star$. This paves the road to prove dissipativity.

**Lemma 5** (*Invariance of Soft Cross-Entropy*)**.** *The softmax probabilities and the soft cross-entropy are invariant to the transformation* $T : \mathbb{X} \to \mathbb{X}$ *of the form*

$$T = I^C - \frac{1}{C}\mathbf{1}^C\mathbf{1}^{C\top}, \tag{14}$$

*where* $I^C \in \mathbb{R}^{C\times C}$ *is the identity matrix,* $\mathbf{1}^C \in \mathbb{R}^C$ *denotes the vector of all ones.* □

**Proof.** The transformation $T$ subtracts the average of all components from the activation vector before applying the softmax. Since the softmax probabilities are invariant to adding a scalar to the activation vector the resulting loss function is also invariant to $T$. A proof of this result can be found, e.g., in Goodfellow et al. (2016, Section 6.2.2.3). Here, for the sake of self-containedness, we include a sketch of the proof. For all classes $y = 1,\dots,C$, the translated softmax probability is invariant to (14), as

$$p(y|Tx) = \frac{e^{x_y - x_{\mathrm{avg}}}}{\sum_{j=1}^{C} e^{x_j - x_{\mathrm{avg}}}} = \frac{e^{x_y}}{\sum_{j=1}^{C} e^{x_j}} = p(y|x),$$

where $x_{\mathrm{avg}} = \frac{1}{C}\mathbf{1}^{C\top}x$ is the average of all components of $x$. The soft cross-entropy is a function of the invariant softmax probabilities and therefore also invariant to the transformations (14). ■

Due to the translational invariance of the classification of Lemma 5, we further restrict the dissipativity analysis in the subspace $\mathbb{X}_T$ where $\mathbf{1}^{C\top}x = 0$ as follows

$$\mathbb{X}_T \doteq \left\{x \in \mathbb{R}^C \,|\, \mathbf{1}^{C\top}x = 0\right\}.$$

The intersection of $\mathbb{X}_T$ and $\mathbb{X}_y^\star$ is given by the point

$$x_y^\star = \arg\min_{\tilde{x}\in\mathbb{X}_T} \tilde{\ell}_{\mathrm{f}}(\tilde{x},y),$$

which is component-wise defined as

$$[\tilde{x}_y^\star]_i = \begin{cases} -\frac{C-1}{C}\delta & i = y \\ \frac{1}{C}\delta & i \neq y. \end{cases}$$

The projection $T : \mathbb{X} \to \mathbb{X}_T$, maps a point $x \in \mathbb{X}$ to $\tilde{x} \in \mathbb{X}_T$, while keeping the value of the soft cross-entropy constant.

**Lemma 6** (*$T$ Preserves the Distance to $\mathbb{X}_y^\star$*)**.** *Consider the transformation $T$ from (14), we have for all $x \in \mathbb{R}^C$ and all labels $y \in \mathbb{N}_{[1,C]}$*

$$\mathrm{dist}(x,\mathbb{X}_y^\star) = \mathrm{dist}(Tx,\mathbb{X}_y^\star), \quad \square$$

**Proof.** Let $x_y^\star$ be the closest point to $x \in \mathbb{R}^C$ in $\mathbb{X}_y^\star$, then also $x_y^\star + \alpha\mathbf{1}^C \in \mathbb{X}_y^\star$, $\forall\alpha \in \mathbb{R}$, so $d = x - x_y^\star$, $d \perp \mathbf{1}^C$. Therefore, $d$ lies in the eigenspace of $T$ corresponding to the eigenvalue 1 and $\tilde{x} - \tilde{x}_y^\star = T(x - x^\star) = x - x^\star$. ■

**Lemma 7** (*Convexity of Soft Cross-Entropy*)**.** *The soft cross-entropy $\tilde{\ell}_{\mathrm{f}} : \mathbb{R}^C \times \mathbb{Y} \to \mathbb{R}^+$, (13), is convex. Moreover, the Hessian of (13)*

$$\tilde{H}(x) = \nabla_x^2\tilde{\ell}_{\mathrm{f}}(x,y). \tag{15}$$

*is independent of the label $y$, positive semi-definite, with 0 as a single eigenvalue and $\mathbf{1}^C$ as the corresponding eigenvector, and all other eigenvalues are positive for all $x \in \mathbb{R}^C$.* □

**Proof.** The Hessian matrix of the cross-entropy

$$\begin{aligned} H(x) = \nabla_x^2\ell(x,y) &= \mathrm{diag}\left(p(x)\right) - p(x)p(x)^\top\right) \\ &= \mathrm{diag}\left((p(x)\right)\left(I^C - \mathbf{1}^C p(x)^\top\right) \end{aligned}$$

is independent of the target probabilities $q(y)$ and therefore equal for the regular and soft cross-entropy $\tilde{H}(x) = H(x)$. The Hessians $H(x)$ and $\tilde{H}(x)$ are positive semi-definite matrices (Singla et al., 2019, Theorem 2).

Moreover, the eigenvalue 0 has the algebraic multiplicity one, since $\mathrm{diag}(p(x))$ is of full rank and therefore $\mathrm{rank}(\tilde{H}(x)) = \mathrm{rank}(I^C - \mathbf{1}^C p(x)^\top)$ with

$$\begin{aligned} \mathrm{rank}\left(I^C - \mathbf{1}^C p(x)^\top\right) &\geq \mathrm{rank}(I^C) - \mathrm{rank}\left(\mathbf{1}^C p(x)^\top\right) = \\ &= C - 1. \end{aligned}$$

The eigenvector of the Hessian corresponding to the eigenvalue 0 is $\mathbf{1}^C$, since $(I^C - \mathbf{1}^C p(x)^\top)\mathbf{1}^C = \mathbf{1}^C - \mathbf{1}^C = 0$ and $p(x)^\top\mathbf{1}^C = 1$. The soft cross-entropy satisfies the second-order characterization of convexity (Bertsekas, 2009, Proposition 1.1.10), i.e., for all $x \in \mathbb{X}$ and all directions $d \in \mathbb{R}^C$,

$$d^\top\tilde{H}(\tilde{x})d \geq 0. \quad ■$$

**Lemma 8** (*Soft Cross-Entropy Lower Bound*)**.** *For the soft cross-entropy (13), there exists $\alpha \in \mathcal{K}_\infty$ such that*

$$\tilde{\ell}_{\mathrm{f}}(x,y) \geq \alpha\left(\mathrm{dist}(x,\mathbb{X}_y^\star)\right). \quad \square \tag{16}$$

**Proof.** Fig. 1 shows the soft cross-entropy for the case of two classes and label $y = 1$ and serves to illustrate the core idea of this proof. The loss function behaves quadratic around the minimizer set before then decreasing in slope and behaving asymptotically linear. Notice that the soft cross-entropy thus behaves similarly to Huber loss (Huber, 1992) used in regression tasks, exhibiting locally quadratic and asymptotically linear behavior to reduce the influence of outliers. The proof is structured as follows. First, we assume that $\tilde{x} \in \mathbb{X}_T$ without loss of generality. Then, we show that the soft cross-entropy $\tilde{\ell}_{\mathrm{f}}(\tilde{x}, y)$ increases with the distance $s$ between the minimizer $\tilde{x}_y^\star$ and the point $\tilde{x} = \tilde{x}_y^\star + s \cdot d \in \mathbb{X}_T$ for all normalized directions $d$ satisfying $\|d\| = 1$. Thirdly, we show the existence of bound that is locally quadratic with the distance $s$. And finally, we show a global lower bound that increases linearly with the distance $s$ and combine it with the quadratic lower bound to create $\alpha \in \mathcal{K}_\infty$.

**Step 1:** Due to Lemmas 5 and 6, both sides of the inequality (16) are invariant to the transformation $T_y$ so without loss of generality we assume that $x = \tilde{x} \in \mathbb{X}_T$. Let $\tilde{x}_y^\star$ be the closest point to $\tilde{x} \in \mathbb{X}_T$ in $\mathbb{X}_y^\star$, then also $\tilde{x}_y^\star + \alpha \mathbf{1}^C \in \mathbb{X}_y^\star$, $\forall \alpha \in \mathbb{R}$, so $d = x - x_y^\star$, $d \perp \mathbf{1}^C$ and therefore $\tilde{x}_y^\star \in \mathbb{X}_T$. Then (16) implies

$$\tilde{\ell}_{\mathrm{f}}(\tilde{x}, y) \geq \alpha \left( \|\tilde{x} - \tilde{x}_y^\star\| \right). \tag{17}$$

Consider the half-line $x_{\mathrm{s}} : \mathbb{R}^+ \times \mathbb{R}^C \times \mathbb{Y} \to \mathbb{X}_T, x_{\mathrm{s}}(s, d, y) = \tilde{x}_y^\star + d \cdot s$. For any direction $d$ with $d \perp \mathbf{1}^C$ and $\|d\| = 1$ the half line $\mathbb{X}_T$ $d \in \mathbb{R}^C$ and $\|x_{\mathrm{s}}(s, d, y) - \tilde{x}_y^\star\| = s$. Consider the soft cross-entropy along the line $x_{\mathrm{s}}(s, d, y)$, $\tilde{\ell}_{\mathrm{f,s}} : \mathbb{R}^+ \times \mathbb{R}^C \times \mathbb{Y} \to \mathbb{R}^+$,

$$\tilde{\ell}_{\mathrm{f,s}}(s, d, y) := \tilde{\ell}_{\mathrm{f}}(x_{\mathrm{s}}(s, d, y), y).$$

Using $\tilde{\ell}_{\mathrm{f,s}}$, the existence of the lower bound (17) can be shown showing that $\tilde{\ell}_{\mathrm{f,s}}(s, d, y) \geq \alpha(s)$ for all $d^\top \mathbf{1}^C$ and $\|d\| = 1$.

**Step 2:** Consider the smallest value of the second derivative $\tilde{\ell}_{\mathrm{f,s}}(s, d, y)$ for $0 \leq s \leq 1$ in any direction $d^\top \mathbf{1}^C$

$$h_{\min} = \min_{\substack{d \in \mathbb{R}^C \\ 0 \leq s \leq 1}} \frac{\mathrm{d}^2 \tilde{\ell}_{\mathrm{f,s}}(s, d, y)}{\mathrm{d}^2 s} = d^\top \tilde{H}(x_{\mathrm{s}}(s, d, y)) d$$
$$\text{s.t.} \quad \|d\| = 1, \qquad \mathbf{1}^{C\top} d = 0.$$

The minimum $\tilde{h}_{\min}$ exists since the constraints form a compact set and is strictly positive, since $d^\top \tilde{H}(x_{\mathrm{s}}(s, y, y)) d > 0$ for $d^\top \mathbf{1}^C = 0$ and $d \neq 0$ (Lemma 7) and is independent of the label. Hence, $\tilde{\ell}_{\mathrm{f,s}}(s, d, y)$ satisfies second-order characterization of strict convexity with respect to $s$ for all $d$ (Bertsekas, 2009, Proposition 1.1.10). Moreover, its first derivative satisfies

$$\frac{\mathrm{d}\tilde{\ell}_{\mathrm{f,s}}(s, d, y)}{\mathrm{d}s} \begin{cases} = 0 & \text{for } s = 0, \\ > 0 & \text{for } s > 0. \end{cases}$$

For $0 \geq s \geq 1$ and all directions $d$, $d^\top \mathbf{1}^C = 0$

$$\tilde{\ell}_{\mathrm{f,s}}(s, d, y) \geq \frac{h_{\min}}{2} s^2$$

is a lower bound.

**Step 3:** Due to the strict convexity of $\tilde{\ell}_{\mathrm{f,s}}(s, d, y)$ the tangent plane at $s = 1$ is a lower bound

$$\begin{aligned} &\tilde{\ell}_{\mathrm{f,s}}(s, d, y) \\ &\geq \tilde{\ell}_{\mathrm{f,s}}(1, d, y) + \frac{\mathrm{d}\tilde{\ell}_{\mathrm{f,s}}(s, d, y)}{\mathrm{d}s}\Big|_{s=1} (s - 1) \\ &\geq \frac{h_{\min}}{2} + \left( \frac{\mathrm{d}\tilde{\ell}_{\mathrm{f,s}}(s, d, y)}{\mathrm{d}s}\Big|_{s=0} + \frac{\mathrm{d}^2 \tilde{\ell}_{\mathrm{f,s}}(s, d, y)}{\mathrm{d}^2 s}\Big|_{s=0} \right) (s - 1) \\ &\geq h_{\min}/2 + h_{\min}(s - 1). \end{aligned}$$

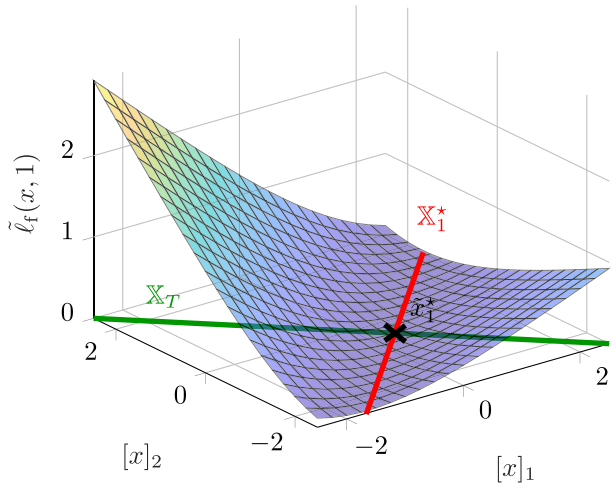


**Fig. 1.** Illustration of the soft cross-entropy and its minimizer set for two classes with the target class $y = 1$.

**Step 4:** Hence, for all $s$, the soft cross-entropy $\tilde{\ell}_{\mathrm{f,s}}(s, d, y)$ for all $d$ with $\mathbf{1}^{C\top} d = 0$ is lower bounded by $\alpha \in \mathcal{K}_\infty$

$$\alpha(s) = \begin{cases} \frac{h_{\min}}{2} s^2 & s \leq 1 \\ h_{\min}(s - 1) + h_{\min}/2 & s > 1 \end{cases} \tag{18}$$

and therefore also (16) holds. ■

### 3.3. Dissipativity in ResNet training

The lower bound to the soft cross-entropy can now be used to formulate a strictly dissipative version of ResNet training.

Based on our previous analysis of the soft cross-entropy, we now focus on the ResNet training OCP (5), with the soft cross-entropy as a stage cost regularization

$$\ell(\mathbf{x}, u) = \tilde{\ell}_{\mathrm{f}}(\mathbf{x}, \mathbf{y}) + r\|u\|^2. \tag{19}$$

**Lemma 9** (*Optimal Steady State*)**.** *The steady state minimizers (9) of the ResNet dynamics (1) with stage cost (19) are given by*

$$\bar{\mathbb{Z}}^\star = \left\{ (\bar{\mathbf{x}}, 0) \in \mathbb{R}^{D \cdot C} \times \mathbb{R}^{n_u} \,\middle|\, \bar{\mathbf{x}} \in \mathbb{X}_{\mathbf{y}}^\star \right\} \tag{20}$$

*with the optimal steady states from*

$$\mathbb{X}_{\mathbf{y}}^\star = \left\{ \begin{bmatrix} x^1 \\ \vdots \\ x^D \end{bmatrix} \in \mathbb{R}^{D \cdot C} \,\middle|\, \begin{matrix} x^d \in \mathbb{X}_{y^d}^\star \\ \forall d = 1, \ldots, D \end{matrix} \right\}. \tag{21}$$

**Proof.** With the ResNet dynamics (1), each state can be rendered a steady state with $u = 0$. Moreover, $u = 0$ is the minimizer of the input penalty $r\|u\|^2$. Therefore, the optimal steady state pairs are of the form $\bar{\mathbb{Z}}^\star = \left\{ (\bar{\mathbf{x}}, 0) \in \mathbb{R}^{D \cdot C} \times \mathbb{R}^{n_u} \middle| \bar{\mathbf{x}} \in \bar{\mathbb{X}}^\star \right\}$. The set of optimal steady states $\bar{\mathbb{X}}^\star = \mathbb{X}_{\mathbf{y}}^\star$ are the minimizers of the soft cross-entropy for each data sample $(x^d, y^d)$, i.e. $x^d \in \mathbb{X}_{y^d}^\star$ are the minimizers of the stage cost $\tilde{\ell}_{\mathrm{f}}(\mathbf{x}, \mathbf{y})$, which correspond to $x^d$. And hence the set optimal steady states is (20).

**Proposition 10** (*Strict Dissipativity*)**.** *The training OCP (5) with stage cost (19) is strictly* $\mathbf{x}$*-*$u$ *dissipative with respect to* $\bar{\mathbb{Z}}^\star$ *from (20).*

*Moreover, the storage function can be chosen as* $\lambda(\mathbf{x}) = c, c \in \mathbb{R}_0^+$. □

**Proof.** The dissipation inequality with constant storage for the stacked state is the summation of the dissipation inequality for the individual data samples

$$\tilde{\ell}_{\mathrm{f}}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{D} \tilde{\ell}_{\mathrm{f}}\left(x^i, y^i\right) \geq \sum_{i=1}^{D} \alpha\left(\operatorname{dist}\left(x^i, \mathbb{X}^\star_{y^i}\right)\right) \geq \alpha(\operatorname{dist}(\mathbf{x}, \mathbb{X}^\star_{\mathbf{y}})),$$

which hold according to Lemma 8. If $r > 0$, then the additional input penalty ensures strict state-input dissipativity $\ell(\mathbf{x}, u) \geq \alpha(\operatorname{dist}((\mathbf{x}, u), \bar{\mathbb{Z}}^\star))$. ■

## 4. Turnpikes in ResNet training

We utilize the dissipativity of the training to analyze the optimal solutions to OCP (5).

Consider the set of timesteps spent $\varepsilon$-close to the optimal state set.

$$\mathcal{Q}_\varepsilon = \left\{k \in \mathbb{N}_{[0,N-1]} \;\middle|\; \operatorname{dist}\left(\mathbf{x}_k, \bar{\mathbb{X}}^\star\right) \leq \varepsilon\right\},$$

and its complement $\hat{\mathcal{Q}}_\varepsilon = \mathbb{N}_{[0,N-1]} \setminus \mathcal{Q}_\varepsilon$.

**Assumption 11** (*Exponential Reachability*)**.** There exists constants $\rho \in [0, 1)$ and $\beta > 0$ and an infinite-horizon control input $\tilde{u} : \mathbb{N}_{[0,\infty)} \to \mathbb{R}^{n_u}$ such that, for all initial conditions $\mathbf{x}^0 \in \mathbf{X}_0 \subseteq \mathbb{R}^{D\cdot C}$, the trajectories satisfies

$$\operatorname{dist}\left(\left(\tilde{\mathbf{x}}_k, \tilde{u}_k\right), \bar{\mathbb{Z}}^\star\right) \leq \beta\rho^k,$$

where $\tilde{\mathbf{x}}_{k+1} = \mathbf{f}_{\mathrm{d}}(\tilde{\mathbf{x}}_k, \tilde{u}_k)$, $\tilde{\mathbf{x}}_0 = \mathbf{x}^0$ is the corresponding state trajectory. □

Note that while initially this assumption seems restrictive, Ruiz-Balet and Zuazua (2023) established a similar reachability for neural ODEs using the ReLU activation function. Furthermore, this assumption implies that there exists a finite network depth that drives the states arbitrarily close to the minimizer set. This ensures that the minimizer set of the correct class is the closest one, which ensures correct classification. However, as no assumptions are made on the test data, this could come at the price of substantial overfitting. How to avoid overfitting in the optimal control approach to deep learning remains an open problem and a question for future research.

**Remark 12** (*Exact Reachability of ResNets*)**.** Consider the ResNet architecture

$$x^i_{k+1} = x^i_k + W_k\sigma\left(A_k x^i_k + b_k\right), \quad x^i_0 = x^i \in \mathbb{R}^C,$$

with controls $W_k \in \mathbb{R}^{C\times C}$, $A_k \in \mathbb{R}^{C\times C}$, and $b_k \in \mathbb{R}^C$ and ReLU activation function, $\sigma(x) = \max\{0, x\}$. Let

$$\mathbb{D} = \left\{\left(x^1, y^1\right), \ldots, \left(x^D, y^D\right)\right\}$$

be the dataset with distinct features $x^i \neq x^j, \forall i, j = 1, \ldots, D$. Then, there exists a finite horizon $N$ and parameters, $W_k$, $A_k$ and $b_k$, such that all samples from the dataset reach the respective set of soft cross-entropy minimizers, i.e. $x^i_N \in \mathbb{X}^\star_{y^i}$ $\forall i = 1, \ldots, D$.

This is based on the reachability result for the corresponding the neural ODE

$$\dot{x}^i(t) = W(t)\sigma\left(A(t)x^i(t) + b(t)\right), \; x^i(0) = x^i, \forall t \in [0, T],$$

see Theorem 2 by Ruiz-Balet and Zuazua (2023). Starting from distinct initial conditions $x^i$ and for any horizon $T$, there exists piecewise constant parameter functions $W(t), A(t), b(t)$, such that all data samples $i \in 1, \ldots, D$ are simultaneously controlled to their distinct targets $x^i(T) = \hat{x}^i$, with $\hat{x}^i \neq \hat{x}^j$ for $i \neq j$. By choosing distinct target points $\hat{x}^i \in \mathbb{X}^\star_{y^i}$, the neural ODE can reach the set of soft cross-entropy minimizers $x^i(T) \in \mathbb{X}^\star_{y^i}$ with piecewise constant parameter functions $W(t), A(t), b(t)$ and at most $6 \cdot D$ switches. By choosing a sufficiently small discretization time $h > 0$, the results transfer to the ResNet on the horizon $N = \left\lceil \frac{T}{h} \right\rceil$, for details we refer to Ruiz-Balet and Zuazua (2023, Remark 3.2). □

**Proposition 13** (*Turnpikes in ResNet Training*)**.** *Consider the training OCP (5) with stage cost (19). Suppose that Assumption 11 holds. Then, there exists a constant $\widehat{V}$ such that, for any parameter $\gamma \in \mathbb{R}^+$ in OCP (5), the optimal solutions satisfy*

$$\#Q_\varepsilon \geq N - \frac{\widehat{V}}{\alpha(\varepsilon)} \qquad \#\hat{Q}_\varepsilon \leq \frac{\widehat{V}}{\alpha(\varepsilon)} \tag{22}$$

*for all $\varepsilon \in \mathbb{R}^+$, where $\#\mathcal{Q}_\varepsilon$ is the cardinality of the set $\mathcal{Q}_\varepsilon$ and*

$$\widehat{V} = 2\left[\beta\frac{1}{1-\rho} + \beta\right]$$

*as the upper bound on the value function for all $\mathbf{x}^0 \in \mathbb{R}^{D\cdot C}$.* □

**Proof.** We first show that the soft cross-entropy loss function is globally Lipschitz with constant $L = 2$, i.e. for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^C$ it holds that

$$\left|\tilde{\ell}_{\mathrm{f}}(\mathbf{x}_1, \mathbf{y}) - \tilde{\ell}_{\mathrm{f}}(\mathbf{x}_2, \mathbf{y})\right| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\| \leq \max_{\tilde{\mathbf{x}}\in\mathbb{R}^{D\cdot C}} \|\nabla_{\tilde{\mathbf{x}}}\tilde{\ell}_{\mathrm{f}}(\tilde{\mathbf{x}}, \mathbf{y})\|\|\mathbf{x}_1 - \mathbf{x}_2\|.$$

The gradient depends on the vector of softmax probabilities $p(x) \in [0, 1]^C$ and target probabilities $\tilde{q}(y) \in [0, 1]^C$. Thus we have

$$\|p(x)\|^2 = \sum_{c=1}^{C} p(c|x)^2 \leq \sum_{c=1}^{C} p(c|x) = 1$$

and likewise for the target probabilities $q(y)$. Hence, the norm of soft cross-entropy gradient is bounded from above by

$$\max_{\tilde{\mathbf{x}}\in\mathbb{R}^{D\cdot C}} \|\nabla_{\tilde{\mathbf{x}}}\tilde{\ell}_{\mathrm{f}}(\tilde{\mathbf{x}}, \mathbf{y})\| \leq \max_{\tilde{x}\in\mathbb{R}^C} \|\|\nabla_{\tilde{x}}\tilde{\ell}_{\mathrm{f}}(\tilde{x}, y)\| = \max_{\tilde{x}\in\mathbb{R}^C} \|p(\tilde{x}) - \tilde{q}(x)\| \leq \max_{\tilde{x}\in\mathbb{R}^C} \|p(\tilde{x})\| + \|\tilde{q}(y)\| \leq 1 + 1 = 2 = L.$$

Hence, the Lipschitz constant of the state cost $\tilde{\ell}_{\mathrm{f}}(\mathbf{x}_k, \mathbf{y}) + r\|u\|^2$ is given by $L_\ell = \max\{r, L\}$. Using this Lipschitz constant, Assumption 11 implies that there exists $\widehat{V}$ such that $V^\gamma_N\left(\mathbf{x}^0\right) \leq \widehat{V}$

$$\begin{aligned} V^\gamma_N\left(\mathbf{x}^0\right) &= \sum_{k=0}^{N-1}\left[\tilde{\ell}_{\mathrm{f}}(\mathbf{x}_k, \mathbf{y}) + r\|u\|^2\right] + \gamma\tilde{\ell}_{\mathrm{f}}(\mathbf{x}_N, \mathbf{y}) \\ &\leq \sum_{k=0}^{N-1} L_\ell \operatorname{dist}\left(\left(\tilde{\mathbf{x}}_k, \tilde{u}_k\right), \bar{\mathbb{Z}}^\star\right) + L_\ell \operatorname{dist}\left(\left(\tilde{\mathbf{x}}_N, \tilde{u}_N\right), \bar{\mathbb{Z}}^\star\right) \\ &\leq L_\ell\left[\sum_{k=0}^{N-1} \beta\rho^k + \gamma\beta\rho^N\right] \leq L_\ell\left[\beta\frac{1}{1-\rho} + \beta\right] =: \widehat{V}. \end{aligned}$$

Then, due to Proposition 10

$$\begin{aligned} &\lambda(\mathbf{x}^\star_N) - \lambda(\mathbf{x}_0) \leq \sum_{k=0}^{N-1} \ell(\mathbf{x}^\star_k, u^\star_k) - \alpha(\operatorname{dist}(\mathbf{z}^\star_k, \bar{\mathbb{Z}}^\star)) \\ \Leftrightarrow\; &\lambda(\mathbf{x}^\star_N) - \lambda(\mathbf{x}_0) + \sum_{k=0}^{N-1} \alpha(\operatorname{dist}(\mathbf{z}^\star_k, \bar{\mathbb{Z}}^\star)) \leq V^\gamma_N\left(\mathbf{x}^0\right) - \gamma\ell_{\mathrm{f}}(\mathbf{z}^\star_N, \mathbf{y}). \end{aligned}$$

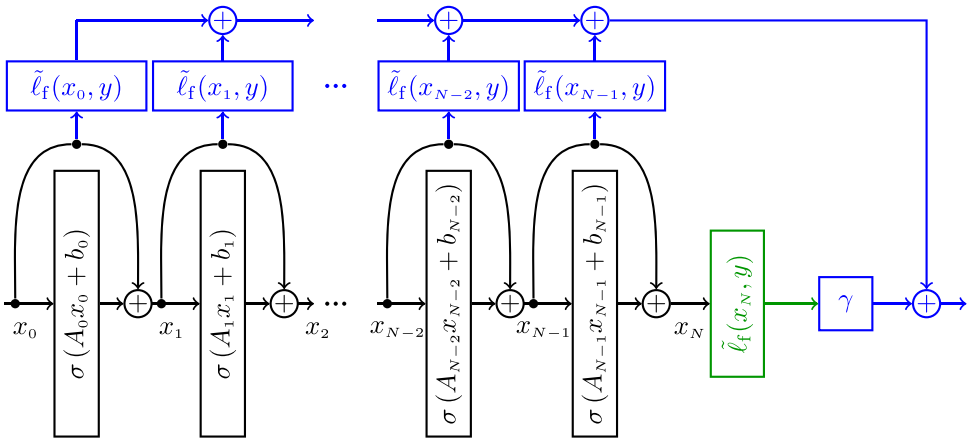


**Fig. 2.** Illustration of the proposed loss function calculation for the ResNet. The propagation of the data through the network depicted in black is unchanged. In addition to the loss function of the output layer in green considered in the standard training formulation, the proposed loss function calculation also evaluates the loss function for the hidden layers in blue. The regularization of the ResNets parameters is also included in the training objective. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

By Proposition 10, we can choose $\lambda(\mathbf{x}) = c, c \in \mathbb{R}_0^+$, and thus $\lambda(\mathbf{x}_N^\star) - \lambda(\mathbf{x}_0) = 0$. Therefore

$$(N - \#\mathcal{Q}_\varepsilon)\alpha_\ell(\varepsilon) \leq \sum_{k=0}^{N-1} \alpha_\ell(\mathrm{dist}(\mathbf{z}_k^\star, \bar{\mathbb{Z}}^\star)) \leq \widehat{V}.$$

Rearranging gives (22). ■

For a singleton optimal steady state-input set $\bar{\mathbb{Z}}^\star = \{\bar{\mathbf{x}}^\star; \bar{u}^\star\}$, this corresponds to Proposition 2 from Faulwasser et al. (2024) or to similar results by Grüne (2013) for generic discrete-time optimal control problems.

Here, however, the turnpike

$$\bar{\mathbb{Z}}^\star = \left\{(\bar{\mathbf{x}}, 0) \in \mathbb{R}^{D\cdot C} \times \mathbb{R}^{n_u} \middle| \bar{\mathbf{x}} \in \mathbb{X}_\mathbf{y}^\star\right\}$$

for the stacked data samples consists of individual subspace turnpikes for each class $\mathbb{X}_1^\star, \ldots, \mathbb{X}_C^\star$ to which the trajectories of samples belonging to the respective class converge.

The optimal trajectories of the training OCP (5) using the soft cross-entropy as stage cost (19) and terminal penalty have a turnpike (Proposition 13) without a leaving arc. Consider the $\varepsilon$ neighborhood of the set of optimal states (21)

$$\mathcal{N}_\varepsilon = \left\{\mathbf{x} \in \mathbb{R}^{D\cdot C} \;\middle|\; \mathrm{dist}\left(\mathbf{x}, \mathbb{X}_\mathbf{y}^\star\right) \leq \varepsilon\right\}.$$

**Remark 14** (*Turnpike Without Leaving Arc*)**.** If the optimal trajectories of (5) enter an $\varepsilon$-neighborhood $\mathcal{N}_\varepsilon$ for some $\varepsilon > 0$, i.e., $\mathbf{x}_{\hat{k}}^\star \in \mathcal{N}_\varepsilon$ for some layer $\hat{k} \in \mathbb{N}_{[0,N]}$, then there exists a layer $\tilde{k} \in \mathbb{N}_{[0,N]}$, $\tilde{k} \geq \hat{k}$, such that for all following layers $k \in \mathbb{N}_{[\tilde{k},N]}$ the optimal trajectories of OCP remain in the $\varepsilon$-neighborhood, i.e. $\mathbf{x}_k^\star \in \mathcal{N}_\varepsilon$. For the ResNet, any state becomes a steady state for zero control inputs, and hence, any neighborhood $\mathcal{N}_\varepsilon$ is rendered forward invariant with zero control input. Thus, the stationary minimizers of the stage cost are the minimizers of the soft cross-entropy loss function together with zero control inputs (Lemma 9). Since, the soft cross-entropy is also used as the terminal penalty, the states from the stationary minimizers also minimize the terminal penalty. Consequently, any leaving arc, which departs from the neighborhood $\mathcal{N}_\varepsilon$, would be suboptimal, c.f. Faulwasser et al. (2024, Proof of Theorem 4). Moreover, if $\mathbf{x}_{\hat{k}}^\star \in \mathbb{X}_\mathbf{y}^\star$, then the optimal trajectories remain at that state, which corresponds to both the minimizer of the stage cost and terminal penalty and thus the following adjoints of OCP (5) become zero, i.e., $\lambda_k = 0$ for all $k \in \mathbb{N}_{[\tilde{k},N]}$ (Faulwasser & Grüne, 2022; Zanon & Faulwasser, 2018).

Table 1 provides an overview of dissipativity properties for the different considered stage costs and regularizations. Fig. 2 illustrates how different choices of the state cost relate to the dissipativity and turnpike phenomenon of the training problem.

## 5. Neural ODEs and other NN architectures

Our preceding analysis has focused on classic ResNets which we conceptualize as discrete-time systems. However, in view of the link between neural ODEs (2) and ResNets (1), we first turn towards the continuous-time extension and then also discuss other architectures.

### 5.1. Continuous-time training formulation

The neural ODE counterpart to (5) reads

$$\min_{u(\cdot)\in\mathcal{L}^\infty([0,T],\mathbb{R}^{n_u})} \int_0^T \ell(\mathbf{x}(t), u(t))\mathrm{d}t + \gamma\,\ell_\mathrm{f}(\mathbf{x}(T), \mathbf{y}) \tag{23a}$$

$$\text{s.t.} \quad \dot{\mathbf{x}}(t) = \mathrm{f_c}(\mathbf{x}(t), u(t)) \quad \forall t \in [0, T], \tag{23b}$$

$$\mathbf{x}(0) = \mathbf{x}^0 \in \mathbb{R}^{C\cdot D}, \tag{23c}$$

where the input signal stacks the bias and weight functions $A(t), b(t)$ and is considered to be in $\mathcal{L}^\infty([0, T], \mathbb{R}^{n_u})$ and the same data-stacking procedure as in Section 2 is applied. The data is propagated over the interval $t \in [0, T]$ and the stacked dynamics are

$$\begin{aligned}\dot{\mathbf{x}}(t) =& \mathrm{f_c}(\mathbf{x}(t), u(t)) \\ \doteq& \sigma\left((I^D \otimes A(t))\mathbf{x}(t) + (\mathbf{1}^D \otimes b(t))\right), \qquad \mathbf{x}(0) = \mathbf{x}^0.\end{aligned} \tag{24}$$

The counterpart to the set $\bar{\mathbb{Z}}$ from (8) reads

$$\bar{\mathbb{Z}}_\mathrm{c} = \left\{(\bar{\mathbf{x}}, \bar{u}) \in \mathbb{R}^{D\cdot C} \times \mathbb{R}^{n_u} \mid 0 = \mathrm{f_c}(\bar{\mathbf{x}}, \bar{u})\right\} \tag{25}$$

Similar to before, optimal steady state pairs are computed via

$$\bar{\mathbf{z}}^\star \in \arg\min_{\bar{\mathbf{z}}} \ell(\bar{\mathbf{z}}) \quad \text{s.t.} \quad \bar{\mathbf{z}} \in \bar{\mathbb{Z}}_\mathrm{c} \tag{26}$$

and the set of all optimal steady states is denoted as $\bar{\mathbb{Z}}_\mathrm{c}^\star \subseteq \bar{\mathbb{Z}}_\mathrm{c}$.

**Definition 15** (*Strict Dissipativity in Cont. Time*)**.** The dynamical system (24) is said to be dissipative with respect to a set of steady-state pairs $\bar{\mathbb{Z}}_\mathrm{c}^\star$ if there exists a non-negative storage function $\lambda : \mathbb{X} \to \mathbb{R}_0^+$ such that for all pairs $\mathbf{z}(t) = (\mathbf{x}(t), u(t))$ defined over some interval $[0, T]$ and all $\bar{\mathbf{z}}^\star \in \bar{\mathbb{Z}}_\mathrm{c}^\star$

$$\lambda(\mathbf{x}(T)) - \lambda(\mathbf{x}(0)) \leq \int_0^T \ell(\mathbf{z}(t)) - \ell(\bar{\mathbf{z}}^\star)\mathrm{d}t. \tag{27a}$$

**Table 1**
Overview of dissipativity properties of ResNet training for different stage costs and regularizations.

| Stage cost | Strict dissipativity | Turnpike object & Comments |
|---|---|---|
| $\ell(\mathbf{x},u)=\|\mathbf{x}-\bar{\mathbf{x}}\|+\|u\|^2$ | Yes | With respect to designed steady states and zero inputs, $\bar{\mathbb{Z}}^\star=\{\bar{\mathbf{x}};0\}$ (Faulwasser et al., 2024). |
| $\ell(\mathbf{x},u)=\ell_{\mathrm{f}}(\mathbf{x},\mathbf{y})+\|u\|^2$ with $\ell_{\mathrm{f}}$ from (7) | Not with $\alpha_\ell\in\mathcal{K}_\infty$ in (10b) | The loss function attains its minimum for $\|\mathbf{x}\|\to\infty$ (Lemma 2), hence any finite horizon trajectory remains arbitrarily far from these states. |
| $\ell(\mathbf{x},u)=\tilde{\ell}_{\mathrm{f}}(\mathbf{x},\mathbf{y})+\|u\|^2$ with $\tilde{\ell}_{\mathrm{f}}$ from (13) | Yes | With respect to the set of soft cross-entropy minimizers and zero inputs $\bar{\mathbb{Z}}^\star=\left\{(\bar{\mathbf{x}},0)\in\mathbb{R}^{D\cdot C}\times\mathbb{R}^{n_u}\,\middle|\,\bar{\mathbf{x}}\in\mathbb{X}_{\mathbf{y}}^\star\right\}$ (Proposition 13). |

If additionally, there exists $\alpha_\ell\in\mathcal{K}_\infty$ such that

$$\lambda(\mathbf{x}(T))-\lambda(\mathbf{x}(0)) \leq \int_0^T \ell(\mathbf{z}(t))-\ell(\bar{\mathbf{z}}^\star)-\alpha_\ell\left(\operatorname{dist}\left(\mathbf{z}(t),\bar{\mathbb{Z}}^\star\right)\right)\mathrm{d}t, \tag{27b}$$

then system (24) is said to be strictly $\mathbf{x}-u$ dissipative with respect to $\bar{\mathbb{Z}}_{\mathrm{c}}^\star$ and for $\mathbf{z}(t)$ replaced by $\mathbf{x}(t)$ in the class $\mathcal{K}$ function the system is said to be strictly $\mathbf{x}$ dissipative.

OCP (23) is said to be (strictly) $\mathbf{x}-u$ dissipative with respect to $\bar{\mathbb{Z}}_{\mathrm{c}}^\star$ if for all $T\in\mathbb{R}_0^+$ and all $\mathbf{x}_0\in\mathbf{X}_0$, the dissipation inequalities 15 hold along any optimal trajectory of (23). □

### 5.2. Continuous-time results

The next corollary to Proposition 10 follows from the observations that the geometry of the soft cross-entropy loss function is not altered in the continuous-time setting and that the entire state space of (24) is covered by steady states with $\bar{u}=0$.

**Corollary 16.** *Consider the training OCP (23) with stage cost (19). OCP is strictly $\mathbf{x}$-$u$ dissipative with constant storage $\lambda(\mathbf{x})=c, c\in\mathbb{R}_0^+$ and with respect to $\bar{\mathbb{Z}}_{\mathrm{c}}^\star=\left\{(\bar{\mathbf{x}},0)\in\mathbb{R}^{D\cdot C}\times\mathbb{R}^{n_u}\,\middle|\,\bar{\mathbf{x}}\in\mathbb{X}_{\mathbf{y}}^\star\right\}$ with $\mathbb{X}_{\mathbf{y}}^\star$ from (21).* □

The set

$$\Theta_{\varepsilon,T}(x_0)\doteq\left\{t\in[0,T]\mid\operatorname{dist}\left(\left(\mathbf{x}^\star(t),\ u^\star(t)\right),\bar{\mathbb{Z}}_{\mathrm{c}}^\star\right)>\varepsilon\right\}$$

collects all time points for which the optimal pairs stay outside of an $\varepsilon$-neighborhood of $\bar{\mathbb{Z}}_{\mathrm{c}}^\star$. The reachability property used for the corresponding turnpike results is stated in the next assumption.

**Assumption 17** (*Exp. Reachability in Cont. Time*)**.** There exists constants $\rho>0$ and $\beta>0$ and an infinite-horizon control input $\tilde{u}(\cdot)\in\mathcal{L}^\infty([0,T],\mathbb{R}^{n_u})$ such that, for all initial conditions $\mathbf{x}^0\in\mathbf{X}_0\subseteq\mathbb{X}$, the trajectories of (24) satisfy

$$\operatorname{dist}\left(\left(\tilde{\mathbf{x}}(t),\tilde{u}(t)\right),\bar{\mathbb{Z}}_{\mathrm{c}}^\star\right)\leq\beta\exp(-\rho t),$$

where $\tilde{\mathbf{x}}(t)$ is the corresponding state trajectory driven by $\tilde{u}(\cdot)$. □

The next result translates Proposition 13 to the continuous-time neural ODE setting.

**Proposition 18** (*Measure Turnpikes*)**.** *Consider the training OCP (23) with stage cost (19). Suppose that 17 holds. Then, there exists a continuous function $\nu:(0,\infty]\to\mathbb{R}_0^+$ independent of $T$ such that, for any $\gamma\in\mathbb{R}$ in OCP (5), the optimal solutions satisfy*

$$\mu[\Theta_{\varepsilon,T}(x_0)]\leq\nu(\varepsilon)<\infty, \tag{28}$$

*where $\mu$ is the Lebesgue measure on the real line.* □

Moreover similar to the ResNet case, the optimal trajectories do no exhibit a leaving arc, similar to Remark 14. The proof follows the usual structure of dissipativity-based proofs of measure turnpikes (Carlson et al., 1991; Faulwasser et al., 2017). It is thus omitted.

### 5.3. Extension to other NN architectures

Our results so far raise the question of whether or not one could consider other architectures than ResNets? To this end, we first formalize the relation between the neural ODEs and the considered NN architecture.

**Definition 19** (*Equilib. Consistent Discretization*)**.** Consider a continuous-time system of the form

$$0=F_{\mathrm{c}}(\dot{x},x,u),\quad x(0)=x^0\in\mathbb{R}^C.$$

Its discretization

$$0=F_{\mathrm{d}}(x_{k+1},x_k,u_k),\quad x_0=x^0\in\mathbb{R}^C$$

is said to be equilibrium consistent, i.e. it preserves the equilibria if any $(\bar{x},\bar{u})$ which solves $0=F_{\mathrm{c}}(0,\bar{x},\bar{u})$ also solves $0=F_{\mathrm{d}}(\bar{x},\bar{x},\bar{u})$ and vice-versa. □

**Proposition 20** (*Dissipativity & Architectures*)**.** *Consider the discrete-time training OCP (5) with the soft-entropy stage cost (19) wherein the considered NN architecture captured in (5b) is an equilibria preserving discretization of (24). Then OCP (5) is strictly dissipative with respect to $\bar{\mathbb{Z}}^\star$ from (20) and the storage function can be chosen as $\lambda(\mathbf{x})=c, c\in\mathbb{R}_0^+$.*

**Proof.** The proof follows from the observation that the structure of the neural ODE (24) combined with $\sigma(0)=0$ implies that for $u=0$ the entire state space of (24) is covered by equilibria. Then applying any equilibria preserving discretization (which may be implicit or explicit, fixed step size of variable step size) means that also in OCP (5) the dynamics (5b) are such that the entire state space is covered by equilibria corresponding to $u=0$. Hence the analysis of the geometry of the soft cross-entropy stage cost (19) can be conducted as in Lemma 7. ■

Turnpike results similar to Proposition 13 are readily inferred if the setting of Proposition 20 is combined with suitable reachability properties. In view of Remark 12 notice that sufficiently accurate equilibria preserving discretizations also preserve reachability, cf. the results of Ruiz-Balet and Zuazua (2023) and Remark 12. Moreover, the previous result shows that in our analysis the crucially helpful requirement on the network architecture is that the entire state space is covered by equilibrium points for zero inputs (which depends on the architecture and on the activation function). Thus, our dissipativity results readily transfer to other ResNet architectures consisting of multiple layers with one skip connection (Esteve & Geshkovski, 2023; He et al., 2016a), e.g., the architecture

$$\begin{aligned} x_{k+1}^i &= x_k^i+\sigma_2\left(A_{k,2}\sigma_1\left(A_{k,1}x_k^i+b_{k,1}\right)+b_{k,2}\right)\\ x_0^i &= x^i\in\mathbb{R}^C, \end{aligned} \tag{29}$$

which first decreases the state dimension to a hidden dimension $h$ by $A_{k,1}\in\mathbb{R}^{h\times C}, b_{k,1}\in\mathbb{R}^h$ and then increases it again by $A_{k,2}\in\mathbb{R}^{C\times h}, b_{k,2}\in\mathbb{R}^C$. It thus has fewer trainable parameters per layer and non-linearity (Esteve et al., 2020). Furthermore, the

dissipativity results also extend to ResNet architectures which include convolutional layers with a skip connection (He et al., 2016a).

**Remark 21** (*Link to Deep Equilibrium Networks*)**.** NN architectures can be obtained from the discretization of neural ODEs or from other considerations. Deep equilibrium networks are based on the observation that the repeated application of a single layer with the same parameters leads to an input-dependent equilibrium. The core idea of equilibrium networks is to obtain this equilibrium using the implicit steady state equation as the model of data propagation via tailored variants of Newton's method. These networks achieve a performance similar the ones with distinct weights across layers, which therefore have many more parameters (Bai et al., 2019; Ling et al., 2024).

In contrast, in the present paper, and in different fashion also in Faulwasser et al. (2024), we shift finding the equilibrium to the training, while the optimal equilibrium subspace (the present paper) or a chosen pre-computed equilibrium (Faulwasser et al., 2024) for the loss function is encoded in the regularization stage cost. In-depth exploration of the links between deep equilibrium networks and our approach is subject to future work.

## 6. Numerical experiments

To validate and illustrate the dissipative formulation of the ResNet training with soft cross-entropy we train networks on the two-spirals task and on the MNIST dataset (Deng, 2012). The training is implemented in Python using the PyTorch framework (Paszke et al., 2019).

### 6.1. Two spirals task

As a first experiment, we consider the two-spirals task classification problem of separating two intertwined spirals (Lang & Witbrock, 1988). The dataset comprising 480 training samples is visualized in Fig. 3. On the dataset, we train a 30-layer ResNet with the architecture (29) with a hidden dimension of $h = 8$, a hyperbolic activation function $\sigma_1(x) = \tanh(x)$ and one identity activation function $\sigma_2(x) = x$. For the label smoothing we use $p_d = 0.95$ as the probability for the correct class. The network is trained using the Adam optimizer (Kingma & Ba, 2014) with a learning rate $\alpha = 0.1$ and weight decay $r = 0.005$ and a terminal penalty $\gamma = 3$.

The evolution of data trajectories in Fig. 4 shows how the two classes are separated in the first ten layers, after which the optimal steady state is reached and the data exhibits the turnpike phenomenon. In addition, the final layer's data trajectories, as shown in Fig. 5, lie closely scattered around the minimizer sets for both classes.

### 6.2. MNIST dataset

To analyze the turnpikes for real-world datasets we consider the MNIST dataset for classifying handwritten digits from 0 to 9 as a second experiment. Each $28 \times 28$ pixel image and is flattened to create the corresponding feature $x^i \in \mathbb{R}^{784}$. We use the ResNet architecture (29) with a hidden dimension of $h = 128$ and one ReLU $\sigma_1(x) = \max\{0, x\}$ and one identity activation function $\sigma_2(x) = x$ and train it according to the OCP (5) with $\gamma = 1$ and $r = 10^{-5}$. For the label smoothing we use $p_d = 0.91$ as the probability for the correct class. The PyTorch optimization hyperparameters are tuned to minimize the training loss, i.e., to foster the visibility of the turnpike phenomenon. The network depth is chosen large to obtain turnpikes.

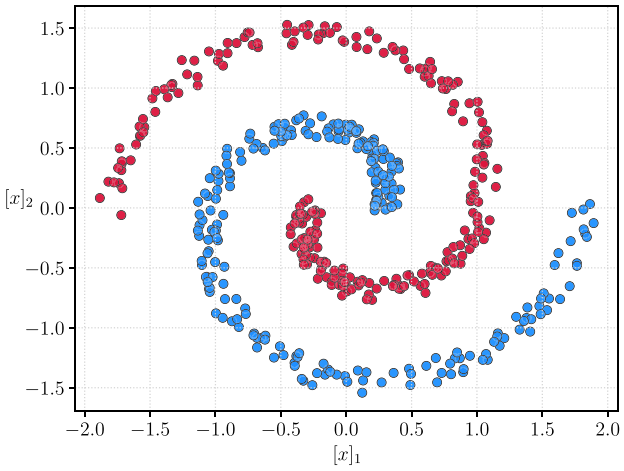


**Fig. 3.** Two Spirals dataset.

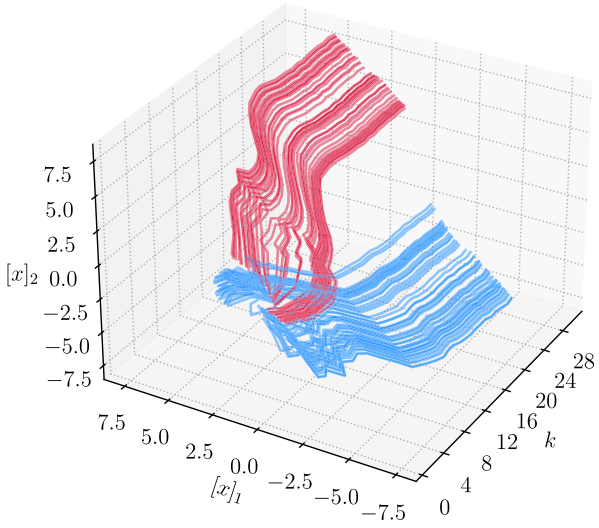


**Fig. 4.** Evolution of the state trajectories for the two classes of the two spirals dataset.

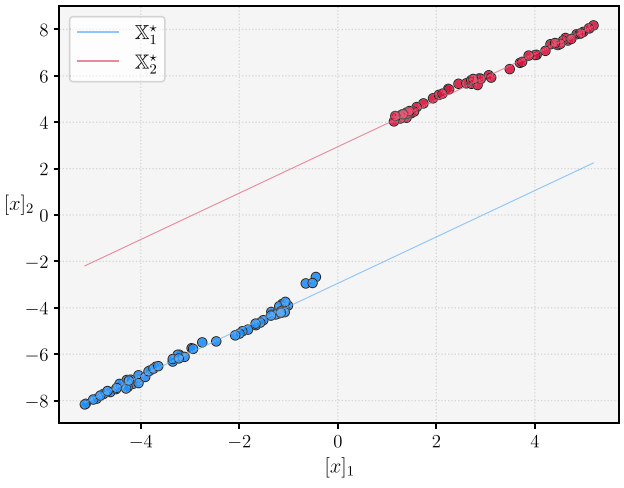


**Fig. 5.** State of the data trajectories in the last layer and the sets of soft cross-entropy minimizers for the two classes, $\mathbb{X}_1^\star$ and $\mathbb{X}_2^\star$.

The softmax probabilities and therefore the loss function are calculated from the first 10 components of each data sample,

see Remark 4. Although the remaining states are not considered in the loss calculation, the weight matrices of the ResNet can still utilize the information contained in these states to aid the classification in the first ten states.

Fig. 6 shows a comparison of training with and without including the soft cross-entropy loss function in the stage cost. It shows the evolution of the training loss over the layers of the ResNet for a 60-layer network trained with the soft cross-entropy in stage cost in comparison to networks trained without the soft cross-entropy stage cost with depths from 10 to 60 layers. The traditional training formulation has a stage cost that only consists of the Tikhonov regularization $\ell(\mathbf{x}, u) = r\|u\|^2$, while the terminal penalty is the soft cross-entropy loss function evaluated for the output of the ResNet, i.e. the terminal state. Fig. 6 illustrates that the networks of different depth trained with this formulation achieve a significant reduction of the soft cross-entropy loss function only after their full depth. Thus, a trained network of a certain depth provides little insight into the performance of a shallower and deeper network which complicates the hyperparamter tuning.

Meanwhile, the 60-layer network trained with the soft cross-entropy regularization achieves a significant reduction in the stage cost in its first 10 layers, after that the stage cost remains at the low level for the next 50 layers indicating the turnpike phenomenon. From a ML perspective, the turnpike spent at an equilibrium state corresponds to an identity mapping, and thus do not contribute to the learned input output transformation and indicates that the first 10 layers are sufficient for the given learning problem. The reaming 50 layers spent at the turnpike can then be removed after the training. The proposed dissipative formulation thus encourages the ResNet to learn the given ML problem in the first layers which indicates the performance of shallower networks. Therefore this formulation includes the automatic exploration of different network depths and their parameters and can be exploited in the search of architectures (Elsken et al., 2019). Starting with conservative guesses for the required network depth, the depth can be reduced if the initial estimate turns out to be too conservative; in this case, the trained network can be pruned and used for inference. Conversely, if the trained network does not exhibit the turnpike phenomenon and the loss continues decreasing over the entire depth of the network, this may indicate that a deeper network might be required.

## 7. Conclusions and outlook

This work has formulated a dissipative version of ResNet training for classification wherein a regularization based on the label-smoothing variant of the cross-entropy is used as stage cost. Assuming asymptotic reachability of the set of minimizers, we prove the existence of subspace turnpikes in the training formulated as an optimal control problem. We have also discussed the extension to neural ODEs and to other NN architectures with skip connections. Experiments on the simple two-spirals task and on the MNIST dataset validate the turnpike results. Our numerical results show that the turnpike phenomenon (without leaving arc) allows to simplify the search of the network depth, since only the first layers strictly necessary for the given classification tasks are used for the transformation between the input and output, the remaining layers can then be pruned.

Follow-up work has already considered the extension to more general neural network architectures (Püttschneider et al., 2025). Yet, the formal analysis of generalization properties of the trained neural network remains open. Moreover, a suboptimality analysis of the interplay of the training algorithm and the problem formulation should be conducted, i.e., an analysis which quantifies the required degree of optimality to observe the turnpike phenomenon.

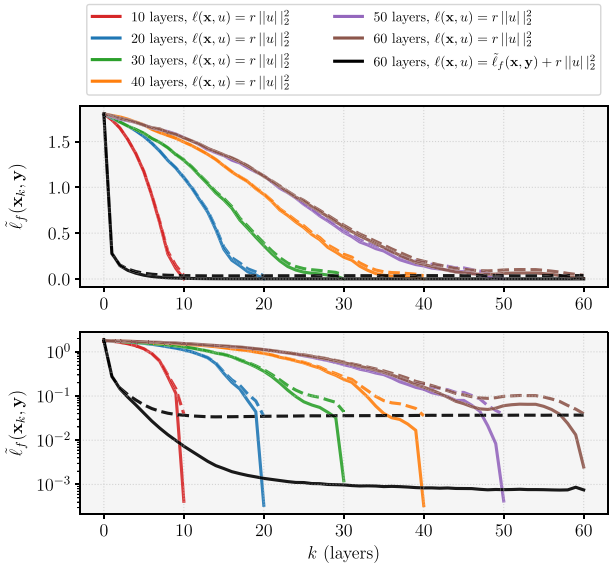


**Fig. 6.** The loss over the layers of the ResNet for the MNIST dataset in linear and logarithmic scale. The straight line represents the training loss and the dashed line represents the test loss.

## References


Angeli, D., Amrit, R., & Rawlings, J. B. (2012). On average performance and stability of economic model predictive control. *IEEE Transactions on Automatic Control*, *57*(7), 1615–1626.

Bai, S., Kolter, J. Z., & Koltun, V. (2019). Deep equilibrium models. In *Advances in neural information processing systems*: *Vol. 32*, (pp. 688–699). Curran Associates, Inc.

Bemporad, A. (2023). Recurrent neural network training with convex loss and regularization functions by extended Kalman filtering. *IEEE Transactions on Automatic Control*, *68*(9), 5661–5668.

Bertsekas, D. P. (2009). *Convex Optimization Theory*. Athena Scientific Belmont.

Bryson, A. E., & Denham, W. F. (1962). A steepest-ascent method for solving optimum programming problems. *Journal of Applied Mechanics*, *29*(2), 247–257.

Carlson, D. A., Haurie, A. B., & Leizarowitz, A. (1991). *Infinite horizon optimal control: deterministic and stochastic systems*. Springer.

Chachuat, B. (2007). EPFL: Nonlinear and dynamic optimization: From theory to practice.

Chang, B., Meng, L., Haber, E., Ruthotto, L., Begert, D., & Holtham, E. (2018). Reversible architectures for arbitrarily deep residual neural networks. In *The 32th annual AAAI conference on artificial antelligence*.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. K. (2018). Neural ordinary differential equations. In *Advances in neural information processing systems*: *Vol. 31*, (pp. 6572–6583). Curran Associates, Inc.

Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *20*(2), 215–232.

Damm, T., Grüne, L., Stieler, M., & Worthmann, K. (2014). An exponential turnpike theorem for dissipative discrete time optimal control problems. *SIAM Journal on Control and Optimization*, *52*(3), 1935–1957.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, *29*(6), 141–142.

Dorfman, R., Samuelson, P., & Solow, R. (1958). *Linear Programming and Economic Analysis*. McGraw-Hill, New York.

Elsken, T., Metzen, J. H., & Hutter, F. (2019). Neural architecture search: A survey. *Journal of Machine Learning Research*, *20*(55), 1–21.

Esteve, C., & Geshkovski, B. (2023). Sparsity in long-time control of neural ODEs. *Systems & Control Letters*, *172*, Article 105452.

Esteve, C., Geshkovski, B., Pighin, D., & Zuazua, E. (2020). Large-time asymptotics in deep learning. arXiv preprint arXiv:2008.02491.

Faulwasser, T., Flaßkamp, K., Ober-Blöbaum, S., Schaller, M., & Worthmann, K. (2022). Manifold turnpikes, trims, and symmetries. *Mathematics of Control, Signals, and Systems*, 1–30.

Faulwasser, T., & Grüne, L. (2022). Turnpike properties in optimal control: An overview of discrete-time and continuous-time results. In E. Zuazua, & E. Trelat (Eds.), *Handbook of numerical analysis*: *Vol. 23*, (pp. 367–400). Elsevier.

Faulwasser, T., Hempel, A.-J., & Streif, S. (2024). On the turnpike to design of deep neural networks: Explicit depth bounds. *IFAC Journal of Systems and Control*, *30*, Article 100290.
Faulwasser, T., & Kellett, C. (2021). On continuous-time infinite horizon optimal control – dissipativity, stability and transversality. *Automatica*, *134*, Article 109907.
Faulwasser, T., Korda, M., Jones, C. N., & Bonvin, D. (2014). Turnpike and dissipativity properties in dynamic real-time optimization and economic MPC. In *53rd IEEE conference on decision and control* (pp. 2734–2739).
Faulwasser, T., Korda, M., Jones, C., & Bonvin, D. (2017). On turnpike and dissipativity properties of continuous-time optimal control problems. *Automatica*, *81*, 297–304.
Feng, Z., & Lam, J. (2011). Stability and dissipativity analysis of distributed delay cellular neural networks. *IEEE Transactions on Neural Networks*, *22*(6), 976–981.
Galimberti, C. L., Furieri, L., Xu, L., & Ferrari-Trecate, G. (2023). Hamiltonian deep neural networks guaranteeing nonvanishing gradients by design. *IEEE Transactions on Automatic Control*, *68*(5), 3155–3162.
Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
Griewank, A. (2012). Who invented the reverse mode of differentiation? *Documenta Mathematica, Extra Volume ISMP*, 389–400.
Grüne, L. (2013). Economic receding horizon control without terminal constraints. *Automatica*, *49*(3), 725–734.
Grüne, L. (2022). Dissipativity and optimal control: Examining the turnpike phenomenon. *IEEE Control Systems Magazine*, *42*(2), 74–87.
Grüne, L., & Müller, M. A. (2016). On the relation between strict dissipativity and turnpike properties. *Systems & Control Letters*, *90*, 45–53.
He, K., Zhang, X., Ren, S., & Sun, J. (2016a). Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition*.
He, K., Zhang, X., Ren, S., & Sun, J. (2016b). Identity mappings in deep residual networks. In *European conference on computer vision*.
Huber, P. J. (1992). Robust estimation of a location parameter. In *Breakthroughs in statistics: methodology and distribution* (pp. 492–518). Springer.
Karsai, A. (2024). Manifold turnpikes of nonlinear port-Hamiltonian descriptor systems under minimal energy supply. *Mathematics of Control, Signals, and Systems*, 1–22.
Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. *International Conference on Learning Representations*.
Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*: *Vol. 25*, (pp. 1097–1105). Curran Associates, Inc.
Lang, K., & Witbrock, M. (1988). Learning to tell two spirals apart. In *Proceedings of the 1988 connectionist models summer school*.
LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.
Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.
Li, Q., Chen, L., Tai, C., & Weinan, E. (2017). Maximum principle based algorithms for deep learning. *Journal of Machine Learning Research*, *18*(1), 5998–6026.
Ling, Z., Li, L., Feng, Z., Zhang, Y., Zhou, F., Qiu, R. C., & Liao, Z. (2024). Deep equilibrium models are almost equivalent to not-so-deep explicit models for high-dimensional Gaussian mixtures. arXiv preprint arXiv:2402.02697.
Martin, T., Köhler, P. N., & Allgöwer, F. (2019). Dissipativity and economic model predictive control for optimal set operation. In *2019 American control conference* (pp. 1020–1026).
Martinelli, D., Galimberti, C. L., Manchester, I. R., Furieri, L., & Ferrari-Trecate, G. (2023). Unconstrained parametrization of dissipative and contracting neural ordinary differential equations. In *2023 62nd IEEE conference on decision and control* (pp. 3043–3048). IEEE.
McKenzie, L. (1976). Turnpike theory. *Econometrica: Journal of the Econometric Society*, *44*(5), 841–865.
Müller, M. A. (2021). Dissipativity in economic model predictive control: Beyond steady-state optimality. In T. Faulwasser, M. A. Müller, & K. Worthmann (Eds.), *Recent advances in model predictive control: theory, algorithms, and applications* (pp. 27–43). Cham: Springer International Publishing.
Müller, R., Kornblith, S., & Hinton, G. E. (2019). When does label smoothing help? In *Advances in neural information processing systems*: *Vol. 32*, (pp. 4696–4705). Curran Associates, Inc.
Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., .... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*: *Vol. 32*, (pp. 8024–8035). Curran Associates, Inc.
Püttschneider, J., Heilig, S., Fischer, A., & Faulwasser, T. (2025). Towards an optimal control perspective of ResNet training. In *High-dimensional learning dynamics 2025*.
Rahmath, H., Srivastava, V., Chaurasia, K., Pacheco, R. G., & Couto, R. S. (2024). Early-exit deep neural network - a comprehensive survey. *ACM Computing Surveys*, *57*(3).
Ramsey, F. P. (1928). A mathematical theory of saving. *The Economic Journal*, *38*(152), 543–559.
Rawlings, J. B., Mayne, D. Q., & Diehl, M. (2019). *Model predictive control: theory, computation, and design* (2nd ed.). Nob Hill Publishing.
Revay, M., Wang, R., & Manchester, I. R. (2023). Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness. *IEEE Transactions on Automatic Control*.
Ruiz-Balet, D., & Zuazua, E. (2023). Neural ODE control for classification, approximation, and transport. *SIAM Review*, *65*(3), 735–773.
Sakamoto, N., & Zuazua, E. (2021). The turnpike property in nonlinear optimal control—A geometric approach. *Automatica*, *134*, Article 109939.
Schaller, M., Philipp, F., Faulwasser, T., Worthmann, K., & Maschke, B. (2021). Control of port-Hamiltonian systems with minimal energy supply. *European Journal of Control*, *62*(7), 33–40.
Singhal, S., & Wu, L. (1988). Training multilayer perceptrons with the extended Kalman algorithm. In *Advances in neural information processing systems*: *vol. 1*, (pp. 133–140). Morgan-Kaufmann.
Singla, S., Wallace, E., Feng, S., & Feizi, S. (2019). Understanding impacts of high-order loss approximations and features in deep learning interpretation. In *International conference on machine learning* (pp. 5848–5856). PMLR.
Speelpenning, B. (1980). *Compiling fast partial derivatives of functions given by algorithms* (Ph.D. thesis), University of Illinois at Urbana-Champaign.
Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *2016 IEEE conference on computer vision and pattern recognition* (pp. 2818–2826).
Teerapittayanon, S., McDanel, B., & Kung, H. (2016). Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition* (pp. 2464–2469).
Trélat, E. (2023). Linear turnpike theorem. *Mathematics of Control, Signals, and Systems*, *35*(3), 685–739.
Trélat, E., & Zuazua, E. (2015). The turnpike property in finite-dimensional nonlinear optimal control. *Journal of Differential Equations*, *258*(1), 81–114.
von Neumann, J. (1938). Über ein ökonomisches gleichungssystem und eine verallgemeinerung des brouwerschen fixpunktsatzes. In *Ergebnisse eines mathematischen seminars*.
Wang, Q., Ma, Y., Zhao, K., & Tian, Y. (2020). A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, 1–26.
Willems, J. C. (1971). Least squares stationary optimal control and the algebraic Riccati equation. *IEEE Transactions on Automatic Control*, *16*(6), 621–634.
Willems, J. C. (1972). Dissipative dynamical systems part i: general theory. *Archive for Rational Mechanics and Analysis*, *45*(5), 321–351.
Zanon, M., & Faulwasser, T. (2018). Economic MPC without terminal constraints: Gradient-correcting end penalties enforce asymptotic stability. *Journal of Process Control*, *63*, 1–14.
Zeng, H.-B., Park, J. H., Zhang, C.-F., & Wang, W. (2015). Stability and dissipativity analysis of static neural networks with interval time-varying delay. *Journal of the Franklin Institute*, *352*(3), 1284–1295.

**Jens Püttschneider** is a Ph.D. student at the Institute of Energy Systems, Energy Efficiency and Energy Economics at TU Dortmund University. He has studied Electrical Engineering and Information Technology at the same university and received his bachelor degree in 2018 and his M.Sc. in 2021

His current research interests are optimization-based control and its connections to deep learning.

**Timm Faulwasser** is a full professor in the School of Electrical Engineering, Computer Science and Mathematics at Hamburg University of Technology, while before he held a professorship at TU Dortmund University. He has studied Engineering Cybernetics with minor in philosophy at the University of Stuttgart (2000–2006). After doctoral studies in the International Max Planck Research School for Analysis, Design and Optimization in Chemical and Biochemical Process Engineering Magdeburg he obtained his Ph.D. from the Department of Electrical Engineering and Information Technology at Otto-von-Guericke-University Magdeburg, Germany in 2012. He has been postdoctoral researcher at École Polytechnique Fédérale de Lausanne (2013–2016) and senior researcher at Karlsruhe Institute of Technology (2015–2019). Previously, Timm was a member of the IEEE-CSS Conference Editorial Board and associate editor of the European Journal of Control as well as IEEE Control System Letters. Currently, he serves as associate editor for the IEEE Transactions on Automatic Control and Mathematics of Control Systems and Signals. He received the 2021–2023 Automatica Paper Prize and the European Control Award 2025.

His current research interests are optimization-based and data-driven control of stochastic and nonlinear systems as well as systems and control approaches to learning.